IN THE CLAIMS

Please amend the claims as follows:

1-45. (Canceled)

46. (Previously Presented) A processor including processor model specific hardware and platform chipset hardware, the processor comprising:

error hardware to correct errors in a first category while the processor continues to execute a current process;

a non-volatile memory;

a first layer of firmware stored in the non-volatile memory to execute a first abstraction layer and including a first error-handling routine to correct at least some errors in a second category;

a second layer of firmware stored in the non-volatile memory to execute a second abstraction layer above the first abstraction layer. the second abstraction layer including a second error-handling routine, communicating with the first error-handling routine, to correct other errors in the second category.

47. (Previously Presented) A processor according to claim 46 where the first abstraction layer is a processor abstraction layer (PAL) that encapsulates the processor model specific hardware.

48. (Previously Presented) A processor according to claim 47 where the PAL interfaces directly to the processor hardware.

49. (Previously Presented) A processor according to claim 47 where the second abstraction layer is a system abstraction layer (SAL) that isolates an operating system from the platform chipset hardware.

50. (Previously Presented) A processor according to claim 47 where the system abstraction layer interfaces directly to the processor abstraction layer.

51. (Previously Presented) A processor according to claim 46 further comprising an error log to store data concerning errors in at least one of the categories.

52. (Currently Amended) A processor according to claim 51 where the error log stores data concerning errors in multiple ones of the categories.[[.]]

53. (Previously Presented) A system, comprising:

a processor including processor model specific hardware and platform chipset hardware, the processor including

error hardware to correct errors in a first category while the processor continues to execute a current process,

a non-volatile memory,

a first layer of firmware stored in the non-volatile memory to execute a first abstraction layer and including a first error-handling routine to correct at least some errors in a second category,

a second layer of firmware stored in the non-volatile memory to execute a second abstraction layer above the first abstraction layer, the second abstraction layer including a second error-handling routine, communicating with the first error-handling routine to correct other errors in the second category;

a system memory;

an operating system stored in the system memory, the operating system including a third error-handling routine, communicating with the second error-handling routine to correct additional errors in a third category.

54. (Previously Presented) A system according to claim 53 where the additional errors are not correctable by the second error-handling routine.

55. (Previously Presented) A system according to claim 53 where the third error handling routine is adapted to terminate a currently running process.

56. (Previously Presented) A system according to claim 53 further comprising an error log adapted to store data concerning errors in the first, second, and third categories.

57. (Previously Presented) A method for execution in a digital processor, the method comprising:

generating a machine check abort condition after detecting an error;

if the error is in a first category, correcting the error in hardware while the processor continues executing a current process;

if the error is in a second category,

interrupting the current process,

attempting to correct the error within a first level of firmware that is adapted to execute a first abstraction layer in the processor;

if the error cannot be corrected within the first level of firmware,

attempting to correct the error in a second level of firmware that is adapted to execute a second abstraction layer interfaced to the first level of firmware;

if the error can be corrected within either level of firmware, resuming the process.

58. (Previously Presented) A method according to claim 57 further comprising saving state information while attempting to correct the error within the first level of firmware.

59. (Previously Presented) A method according to claim 58 further comprising saving additional state information while attempting to correct the error within the second level of firmware.

60. (Previously Presented) A method according to claim 57 further comprising logging data concerning the error.

61. (Previously Presented) A method according to claim 60 where the error data is logged while attempting to correct the error in hardware.

62. (Previously Presented) A method according to claim 57 further comprising terminating the current process if the error cannot be corrected within the second level of firmware.

63. (Previously Presented) A method according to claim 62 further comprising, if the error is in a third category, attempting to correct the error within software associated with an operating system.

64. (Currently Amended) A computer memory ~~medium bearing~~ storing instructions, ~~and data that causes a suitably programmed digital computer to execute~~ the instructions operable to be executed on a computer to perform a method comprising:

generating a machine check abort condition after detecting an error;

if the error is in a first category, correcting the error in hardware while the processor continues executing a current process;

if the error is in a second category,

interrupting the current process,

attempting to correct the error within a first level of firmware that is adapted to execute a first abstraction layer in the processor;

if the error cannot be corrected within the first level of firmware,

attempting to correct the error in a second level of firmware that is adapted to execute a second abstraction layer interfaced to the first level of firmware;

if the error can be corrected within either level of firmware, resuming the process.

65. (Currently Amended) ~~A medium according to~~ The computer memory of claim 64 where the method further comprises terminating the current process if the error cannot be corrected within the second level of firmware.

66. (Currently Amended) ~~A medium according to~~ The computer memory of claim 64 further comprises, if the error is in a third category, attempting to correct the error within software associated with an operating system.

67. (Previously Presented) A method executing on at least one digital processor executing at least one current process, the method comprising:

generating a machine check abort condition after detecting an error;

if the error has a first category, attempting to correct the error in hardware associated with the processor, without interrupting the processor;

if the error has a second category, attempting to correct the error in firmware associated with the processor, without terminating the current process;

if the error has a third category, attempting to correct the error in operating system software associated with the processor.

68. (Previously Presented) A method according to claim 67 further comprising, if the error has the third category, terminating the current process.

69. (Previously Presented) A method according to claim 67 further comprising, if the error has a fourth category, halting or rebooting the processor.

70. (Previously Presented) A method according to claim 67 further comprising logging data concerning the error during all of the attempting operations.

71. (Previously Presented) A method according to claim 67 where attempting to correct the error in firmware comprises:

attempting to correct the error in a first layer of the firmware, the first layer being adapted to encapsulate model specific hardware of the processor;

if the error is not corrected in the first layer of the firmware, attempting to correct the error in a second layer of the firmware, the second layer being adapted to isolate platform chipset hardware of the processor from an operating system.

72. (Currently Amended) A data-processing system comprising:

a plurality of processing elements each adapted to execute one of a plurality of concurrent processes and to signal a machine check abort condition to others of the processing elements after an error occurs in the each processing element;

a non-volatile memory accessible to a first of the processing elements,

a firmware error handler stored in the non-volatile memory and responsive to the machine check abort signal to cause the first of the processing elements to attempt to correct the error in any of the processing elements.[[.]]

73. (Previously Presented) A system according to claim 72 further comprising a log to store data concerning the machine check abort condition arising form any of the processing elements.

74. (Previously Presented) A system according to claim 72 where the error handler comprises:

a PAL error handler to create an entry in an error log, to save state information, and to attempt to correct the error;

a SAL error handler to read the entry in the error log and to attempt to correct the error, in response to failure of the PAL error handler to correct the error.

75. (Previously Presented) A system according to claim 72 further comprising:

a system memory;

an operating system stored in the system memory and including an OS error handler to read the entry in the error log and to attempt to correct the error in response to failure of the firmware error handler.

76. (Previously Presented) A system according to claim 75 where the OS error handler terminates at least one of the concurrent processes.

77. (Previously Presented) A system according to claim 72 where the machine check abort signal idles all of the processing elements except the first processing element.

78. (Previously Presented) A method comprising:

generating a machine check abort signal indicating an error in one of a plurality of processing elements;

if the error is a local error,

attempting to correct the error within the one processing element by an error handler in firmware that implements an abstraction layer of the one processing element;

if the error is a global error,

communicating the machine check abort signal to a monarch processing element of the plurality of processing elements, the monarch processing element being different from the one processing element in which the error occurred;.

attempting to correct the error by an error handler within the monarch processor.

79. (Previously Presented) A method according to claim 78 further comprising idling the remaining processing elements while the monarch processor is attempting to correct the error.

80. (Previously Presented) A method according to claim 78 where the monarch processor attempts to correct the error only for a rendezvous state of the plurality of processing elements.

81. (Previously Presented) A method according to claim 80 where the rendezvous state occurs if the error is both global and severe.

82. (Previously Presented) A method according to claim 81 where the rendezvous state occurs if a predetermined number of errors have occurred within a predetermined time.

83. (Previously Presented) A method according to claim 78 where

the one processing element creates an entry in a log containing data concerning the error;

the monarch processor reads the entry in the log while attempting to correct the error.

84. (Previously Presented) A method according to claim 78 where

the one processing element saves state information;

the monarch processing element reads the state information while attempting to correct the error.

85. (Previously Presented) A method according to claim 78 where the monarch attempts to correct the error by an error handler in firmware that implements an abstraction layer of the monarch processing element.

86. (Currently Amended) A computer memory ~~medium bearing~~ storing instructions, ~~and data that causes a suitably programmed digital computer to execute~~ the instructions operable to be executed on a computer to perform a method comprising:

generating a machine check abort signal indicating an error in one of a plurality of processing elements;

if the error is a local error,

attempting to correct the error within the one processing element by an error handler in firmware that implements an abstraction layer of the one processing element;

if the error is a global error,

communicating the machine check abort signal to a monarch processing element of the plurality of processing elements, the monarch processing element being different from the one processing element in which the error occurred;.

attempting to correct the error by an error handler within the monarch processor.

87. (Currently Amended) ~~A medium according to~~ The computer memory of claim 86 where the monarch processor attempts to correct the error only for a rendezvous state of the plurality of processing elements.

88. (Currently Amended) ~~A medium according to~~ The computer memory of claim 86 where the monarch attempts to correct the error by an error handler in firmware that implements an abstraction layer of the monarch processing element.